AMENDMENTS TO THE CLAIMS

This listing of claims replaces all previous versions and listings of claims in this application.

Claim Listing:

1. (Currently amended) A method for decreasing the latency between an instruction cache and a pipeline processor having a plurality of parallel execution stages, each execution stage having a decode stage and an instruction queue for sequentially processing instructions being processed by said processor, comprising:

determining whether said decode stage and said instruction queue do not have valid data; and

simultaneously inserting instructions from said instruction cache in parallel to said decode stage and instruction queue when said decode stage and said instruction queue contain invalid data.

2. (Currently amended) ~~A method for decreasing the latency between an instruction cache and said pipeline processor according to~~ The method of claim 1 further comprising:

processing said cache instructions from said cache sequentially through said decode stage and said instruction queue when valid data exists in said instruction queue.

3. (Currently amended) A method for processing instructions in a pipelined processor having a series of pipelined stages which reduces latency between an instruction queue and a pipeline processor, the method comprising:

serially fetching a plurality of instructions to be executed in said pipeline processor from a cache memory;

decoding each of said fetched ~~instruction addresses~~ plurality of instructions in a first stage of said pipeline processor to determine if an execution branch is to be taken;

simultaneously loading one or more of said ~~instruction~~ fetched plurality of instructions into said instruction queue ~~at the same time said instruction is being loaded in~~ and into said decoder when said instruction queue and decoder are both empty;

sequentially loading each of said ~~instruction~~ fetched instructions into said instruction queue from said decoder when said instruction queue and said decoder are not empty; and

shifting ~~the~~ contents of ~~an~~ said instruction queue to produce an instruction from said instruction queue for processing in subsequent pipeline stages.

4. (Currently amended) The method ~~for processing instructions in a pipelined processor according to~~ of claim 3, wherein said decoder identifies each of said plurality of fetched instructions loaded in said instruction queue at the same time as said instructions which are loaded in said decoder as valid or invalid during a subsequent cycle of said pipeline processor if an execution branch is not taken.

5. (Original) The method for processing instructions in a pipelined processor according to claim 3 wherein said instruction queue contents are shifted left to an output port connected to plural pipelined processor stages.

6. (Currently amended) A method for executing instructions in a pipelined processor, the method comprising:

sequentially fetching ~~the addresses of~~ instructions to be executed by said pipelined processor;

determining if said fetched instructions are stored in a cache memory;

determining whether both a decode stage of a decoder and ~~location of~~ an instruction queue stage of an instruction queue of said ~~pipe-line~~ pipelined processor ~~is~~ are empty;

simultaneously loading said fetched instructions from said cache memory ~~in~~ into said decode stage and into said instruction queue stage in parallel when both said ~~stages~~ decode stage and said instruction queue stage are empty; and

sequentially reading out instructions from said instruction queue ~~instructions~~ for execution in said pipelined processor.

7. (Currently amended) The method ~~for executing instructions in a pipelined processor according to~~ of claim 6, further comprising:

loading only said decode stage with said instructions when said instruction queue stage contains valid data, and sequentially transferring said instructions to said instruction queue when a position in said instruction queue is available.

8. (Currently amended) The method ~~for executing instructions in a pipelined processor according to~~ of claim 7, further comprising:

identifying one of said fetched ~~instruction~~ instructions as a branch instruction if said decoder predicts that a branch is being taken ~~from said instruction~~; and

inhibiting transfer of subsequent instructions from said decoder to said instruction queue.

9. (Currently amended) The method ~~for executing instructions in a pipeline processor according to~~ of claim 7, further comprising:

fetching an instruction from a main memory when said branch instruction is not in said cache memory; and

forwarding said fetched instruction to said decode stage for sequential transfer to said instruction queue.

10. (Currently amended) The method ~~for executing instructions in a pipeline processor according to~~ of claim 6, further comprising:

~~examining the contents of a location in said queue; and~~

determining ~~the~~ a state of a valid bit in ~~each of said locations whereby the determination of~~ a location in said instruction queue so as to determine whether said location contains valid data ~~is made~~.

11. (Original) The method for executing instructions in a pipeline processor according to claim 7, wherein said instructions are transferred from said decode stage to said instruction queue each time an instruction is read from said instruction queue.

12. (Currently amended) An apparatus for reducing the latency between stages of a pipelined processor, the apparatus comprising:

an instruction cache producing a plurality of instructions for execution by said pipelined processor;

a plurality of decode stages connected to receive ~~a~~ the plurality of instructions from said cache;

an instruction queue having a plurality of locations ~~for receiving, wherein one of the plurality of locations is configured to receive~~ an associated instruction and a valid bit; and

~~a plurality of multiplexers, for receiving each of said instructions, an output of a respective decode stage receiving said instructions, and connected to receive a valid bit from a location of said queue as a select signal, said multiplexer connected to supply each of said~~

~~instruction queue locations with one of said instructions selected from either from said decoder stage or from said cache~~

a multiplexer associated with said one of the plurality of locations,

wherein, depending on a value of the valid bit, said multiplexer supplies said one of the plurality of locations in the instruction queue with either an output of one of the decode stages or an output of the instruction cache,

wherein, when the output of the instruction cache is supplied to said one of the plurality of locations in the instruction queue, the output of the instruction cache is simultaneously supplied to said one of the decode stages.

13. (Currently amended) The apparatus according to claim 12, wherein said multiplexer receives a shift signal for said instruction queue which shifts ~~the~~ contents of said instruction queue towards an output port of said instruction queue, and which enables said instructions from said decoder to be transferred to said instruction queue.

14. (Currently amended) The apparatus according to claim 12, wherein said output port of said instruction queue is connected to a plurality of parallel processing stages.

15. (Currently amended) The apparatus according to claim 14, wherein said processing stages execute instructions belonging to one of a load/store operation, an arithmetic operation, or a branch target instruction.